Replicability Stamp

Version 0.1 - October 26, 2016

The *replicability stamp* is an additional recognition for authors of accepted papers who are willing to go one step further, providing a complete implementation of their algorithm, which replicates the results presented in the paper. The replicability stamp is not meant to be a measure of scientific quality of the paper or of the usefulness of presented algorithms. Rather, it is meant to be an endorsement of the replicability of the results presented in the paper and the recognition of the service provided to the community by releasing the code. Submissions for the replicability stamp will be considered only *after* the paper has been fully accepted.

Submissions that are awarded the replicability stamp will receive additional exposure by being listed on the replicability stamp website (http://www.replicabilitystamp.com).

The purpose of this award is to promote reproducibility of research results and to allow scientists and practitioners to immediately benefit from state-of-the-art research results, without spending months re-implementing the proposed algorithms and trying to find the right parameter values. We also hope that it will indirectly foster scientific progress, since it will allow researchers to reliably compare with and to build upon existing techniques, knowing that they are using exactly the same implementation.

This is an initiative supported by a growing list of publishers, journals and conferences.

Application Process Any paper accepted (at any point in time) at one the following journals and venues is eligible to apply for the stamp:

- TODO

.

The application process is lightweight — to apply fill the following google form `https://goo.gl/forms/95W3mrOQWeRGoP392`. You will have to provide general information on the submission, a representative image, a link to a public git repository with the source code, and instructions on how to compile and replicate the results. The code should compile on a vanilla installation of one of the major operating systems (Linux, MacOSX, or Windows), have a license that allows free academic usage, and only depend on libraries that are free for academic use. The code quality will not be evaluated, the purpose of the stamp is only to simplify replicability of the results: in its simplest form, the code should reproduce the data used to generate every result figure shown in the paper.

Detailed Guidelines

To ensure a fair and efficient review of the code submissions, the following guidelines must be followed when you prepare your submission. If there are technical problems that prevent you from complying with the requirements, please contact the chair.

1. Distribution The software must compile and run on *at least one* of the following operating systems, using exactly one of the versions listed below:

- Windows 10 or newer
- Ubuntu 16.04.1 LTS or newer
- MacOSX 10.12 or newer

Note that you can pick whichever operating system you developed your code on. Your code does not need to compile on all of them to obtain the stamp.

The code *must* compile and run on a vanilla installation, without any software installed. If you need specific software to be installed, you should provide a script that automatically downloads and installs all the required dependencies. In the case of Windows, a text file with precise installation instructions will also be accepted.

The software can be written in any programming language, as long as the compiler/interpreter is free to use for academic purposes. One exception is MATLAB, which will be accepted (version 2016b or newer) since it is popular in the graphics community: you can use any standard or custom toolbox and you can have mex functions, as long as they satisfy the license requirements described below.

Detailed instructions to install the compilers/interpreter and the dependencies should be available in a readme file.

2. Reproducing Results An independent script should be provided for every figure or table of the paper that contains a result generated with the algorithm. The script should run without parameters and generate the data that is shown in the figure or listed in the table. If the produced data is not in a standard format, please describe the format in a text file attached to the submission. The script may either directly render the image or generate the data that has been used to produce the image.

The review process will only ensure that the code runs and is able to reproduce the material for the figures. The quality and robustness of the code will not be evaluated and they will have no impact on the review process.

If accurate timings are provided in the paper, there must be a script that produces a text file with a list of all the reported timings. The txt file replicated by the reviewer will become part of the stamp submission — it is of course not necessary for these timings to be identical to the ones reported in the paper.

3. License The code itself and all of its dependencies should have a license that allows free academic use. Note that this does not preclude charging your users for commercial usage (the CGAL project is a successful example of this code distribution model).

For example, you can use Mosek or Gurobi, since they are free for academic purposes, while KNitro cannot be used since their academic license is not free. The only exception to this rule is the usage of MATLAB, due to its popularity in our community. In case your project cannot satisfy this rule, please contact the chair to discuss the special case.

4. Interactive Applications If your application is interactive and it is not possible to script it to replicate the results in the paper, you must provide one short movie clip capturing the screen while you use your software to reproduce the figure. The reviewers will then follow step by step the video to reproduce the same results. If you want to use this option, you will have to provide a short description to justify why it is not possible to script the generation of the results.

5. Hardware requirements We encourage the authors to provide code that runs without requiring a specific hardware, to favor portability and simplify the replicability of the results. However, if this is not possible, we will accept submissions that run on specific hardware as long as this is justified in the submission. The following non-standard hardware is currently allowed: NVIDIA Tesla, NVIDIA GTX 1080, NVIDIA TITAN BLACK, and AMD Firepro D700. If the hardware is not in this list, please contact the chair.

6. Exceptions The previous rules are general guidelines: If your software cannot satisfy one of the previous requirements, please write as soon as possible to the chair to explain the reasons. In many cases, it will be possible to lift any of the previous requirements if there is a good reason for doing it. In particular, there might be cases where the data used in a paper cannot be publicly released: in these cases, the stamp can still be awarded, but the data must be separately sent to the reviewers for the evaluation process.

7. Submission Procedure The submission for the stamp takes place only after the paper is accepted and it is not blind. A single reviewer will be assigned to each submission and the reviewer will work together with the authors to ensure that the submission is improved until it satisfies all the guidelines.

The authors must create a public git repository, commit all the material there and fill the form available at `https://goo.gl/forms/95W3mrOQWeRGoP392`. You will receive a confirmation email after completing the form — if you do not receive the email please contact the chair.

The repository should contain:

- A txt file containing the title of the submission, the authors, and the operating system
- A script that automatically downloads all required dependencies and compiles the code. If this is not possible (for example if MATLAB needs to be installed), then a text file with precise instructions should be provided. The instructions should describe all the software dependencies that need to be installed, together with detailed compilation instructions. Please always specify entire paths: for example, do not write ”change the cmake variable sx to point to the directory where solver x is installed” but instead tell the user to install the solver x in ”c:/solverx” and always use absolute paths (assuming a vanilla installation of the operating system).
- The source code
- A script for every result figure created with the proposed algorithm, which runs the provided implementation and generates the data used to create the figure. In case of a comparison figure, the script only needs to reproduce the parts created with the proposed algorithm. Note that the script does not necessarily need to create a figure: if the figure shows a triangle mesh, it is sufficient to export the triangle mesh into a text file.
- A script that produces a text file with all the timings. (This is only needed if you report timings in the submission).